Robust Head-Pose Estimation Based on Partially-Latent Mixture of Linear Regressions

Vincent Drouard, Radu Horaud, Antoine Deleforge, Sileye Ba, Georgios Evangelidis

Robust Head-Pose Estimation Based on Partially-Latent Mixture of Linear Regressions

Vincent Drouard*, Radu Horaud*, Antoine Deleforge†, Silèye Ba*‡ and Georgios Evangelidis*§
*INRIA Grenoble Rhône-Alpes, Montbonnot Saint-Martin, France
†INRIA Rennes Bretagne Atlantique, Rennes, France
‡VideoStitch, Paris, France §DAQRI Int, Dublin, Ireland

***Abstract*—Head-pose estimation has many applications, such as social event analysis, human-robot and human-computer interaction, driving assistance, and so forth. Head-pose estimation is challenging because it must cope with changing illumination conditions, variabilities in face orientation and in appearance, partial occlusions of facial landmarks, as well as bounding-box-to-face alignment errors. We propose tu use a mixture of linear regressions with partially-latent output. This regression method learns to map high-dimensional feature vectors (extracted from bounding boxes of faces) onto the joint space of head-pose angles and bounding-box shifts, such that they are robustly predicted in the presence of unobservable phenomena. We describe in detail the mapping method that combines the merits of unsupervised manifold learning techniques and of mixtures of regressions. We validate our method with three publicly available datasets and we thoroughly benchmark four variants of the proposed algorithm with several state-of-the-art head-pose estimation methods.**

***Index Terms*—Head pose, face detection, mixture of linear regressions, manifold learning, expectation-maximization.**

I. Introduction

Head pose is an important visual cue in many scenarios such as social-event analysis [1], human-robot interaction (HRI) [2] or driver-assistance systems [3] to name a few. For example, in social-event analysis, 3D head pose information drastically helps to determine the interaction between people and to extract the visual focus of attention [4]. The pose is typically expressed by three angles (pitch, yaw, roll) that describe the orientation with respect to a head-centered frame. The estimation of the pose parameters is challenging for many reasons. Algorithms for head-pose estimation must be invariant to changing illumination conditions, to the background scene, to partial occlusions, and to inter-person and intra-person variabilities. In most application scenarios, faces have small support area, i.e. bounding boxes, typically of the order of 100×100 pixels. Even if the face bounding box is properly detected, one has to extract the pose angles from low-resolution data.

Recent advances in computer vision have shown the relevance of representing an image patch with a feature vector, e.g. SIFT [5], HOG [6], SURF [7], or one of their variants. The rationale of representing faces in a high-dimensional feature space is that the latter supposedly embeds a low-dimensional manifold parameterized by the pose parameters, or the head-pose manifold, e.g. [8]–[10]. Hence, several attempts were carried out in order to cast the problem at hand into various frameworks, such as manifold learning (unsupervised) [11], regression [10], [12], convolutional neural networks [8] (supervised), or dimensionality reduction followed by regression [9], to cite just a few.

While the papers just cited yield interesting and promising results, there are several major issues associated with representing faces with high-dimensional feature vectors, issues that have not been properly addressed. Indeed and as already mentioned, these vectors contain many underlying phenomena other than pose, e.g. illumination, appearance, shape, background, clutter, etc. Hence, one major challenge is to be able to remove these other pieces of information and to retain only head-pose information. Another drawback is that head pose relies on face detection, a process that amounts to finding a bounding box that contains the face and which is invariant to face orientation.

Take for example the case of finding pose parameters using linear or non-linear manifold learning followed by regression. This is usually justified by the fact that high-dimensional-to-low-dimensional (high-to-low) regression has to estimate a very large number of parameters, typically of the order of D^2 where D is the dimension of the feature space. This in turn requires a huge training dataset. Moreover, this sequential way of doing presents the risk to map the input onto an intermediate low-dimensional space that does not necessarily contain the information needed for the finally desired output – head pose. Finally, the estimation of the pose angles in two distinct steps cannot be conveniently expressed in a single optimization formulation.

Supervised pose estimation also requires an annotated dataset of faces with their bounding-box locations and the corresponding pose parameters. The large majority of existing methods relies on manually annotated faces for training. In particular this ensures good bounding-box-to-face alignment, i.e. the face center coincides with the bounding-box center and the bounding box does not contain too many background pixels. This alignment requirement has, however, an important drawback, namely that bounding boxes found by face detection techniques do not correspond to the annotated ones. This means that feature vectors obtained with a face detector do not

This work has been funded by the European Research Council through the ERC Advanced Grant VHIA #340113. R. Horaud acknowledges support from a XEROX University Affairs Committee (UAC) grant (2015-2017).

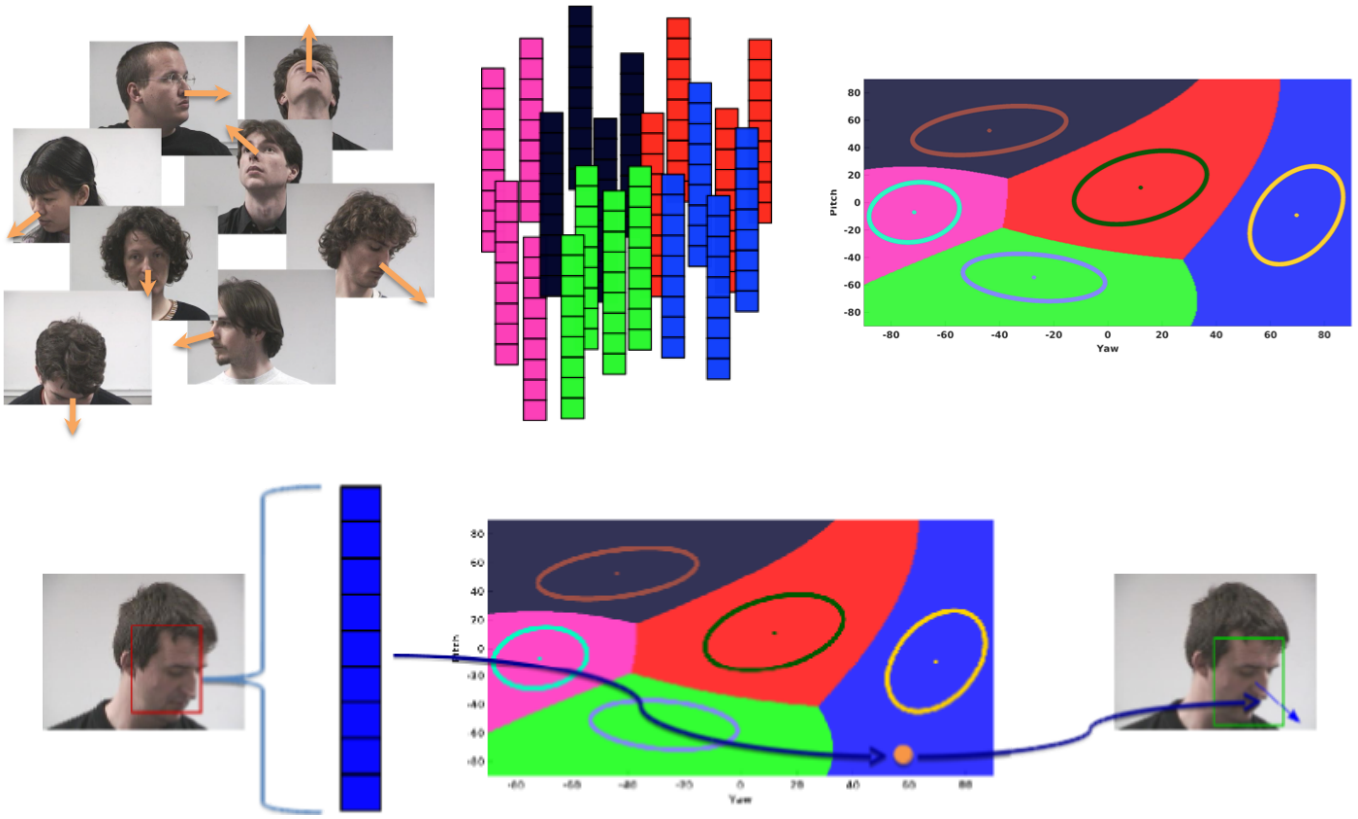

Fig. 1. Pipeline of the proposed supervised head-pose estimation method. **Top:** the parameters of a mixture of linear regressions are learnt from faces annotated with their poses (left). The result of this learning is a simultaneous partitioning of both the high-dimensional input (high-dimensional feature vectors shown in the middle) and low-dimensional output (two-dimensional parameter space shown on the right), such that each region in this partition corresponds to an affine mapping between the input and the output. Moreover, the output is modeled by a Gaussian mixture and each region corresponds to a mixture component. This yields a predictive distribution that can then be used to predict an output from a test input. **Bottom:** A face detector is used to localize a bounding box (left, shown in red) from which a HOG descriptor, namely a high-dimensional feature vector, is extracted. Using the predictive distribution just mentioned, it is then possible to estimate the head-pose parameters (yaw and pitch in this example). Additionally, it is also possible to refine the bounding-box location such that the latter is optimally aligned with the face (right, shown in green).

necessarily belong to the space of feature vectors obtained by manual annotation and used for training. Hence, there is a risk that the head pose, predicted from feature vectors associated with automatic face detection, is incorrectly estimated.

In this paper we propose to learn with both head-pose parameters and bounding-box-to-face alignments, such that, at runtime both the head-pose angles and bounding-box shifts are predicted. This ensures that the bounding-box-to-face alignments predicted with our method are similar with those used for training. Consequently, background variations have minimal influence on the observed feature vector from which the pose is being predicted. This prevents from pose-estimation errors due to discrepancies that occur between manually annotated faces (used for training) and automatically detected faces (used for prediction). We adopt a high-dimensional description of faces based on HOG features, hence we need to solve a high-to-low regression problem. Rather then performing dimensionality reduction and regression in two distinct steps, we propose to use a generative model that unifies these two steps in a single one.

More precisely, we propose to adopt the partially-latent mixture of linear regression model of [10]. When applied to head pose estimation from feature vectors, this method has two advantages: (i) it solves high-dimensional to low-dimensional regression in a single process, without the need of a dimensionality reduction pre-processing step, and (ii) the method incorporates *latent variable augmentation*. More precisely, the output variables are only partially observed such that the regression function can be trained with partially observed outputs. The method adopts an inverse regression strategy [13]: the roles of the input and output variables are interchanged, such that high-to-low regression is replaced with low-to-high regression, followed by high-to-low prediction. As illustrated by our experiments, e.g. Fig. 1, we found that the inclusion of a partially-latent output is particularly relevant whenever the high-dimensional input is corrupted by irrelevant information that cannot be easily annotated, e.g. facial expressions, changes in illumination conditions, changes in background, etc.

Another contribution is a thorough experimental evaluation and validation of the merits of [10] when the task is to recover low-dimensional parameterizations of manifolds embedded in high-dimensional feature vector representations of image patches. We note that this task is ubiquitous in computer vision and image processing, hence a method that unifies dimensionality reduction and regression in a principled manner

is of primary importance and of great utility for practitioners. Moreover, with respect to [10] we provide more detailed mathematical derivations as well as a unified view of the algorithms used for training, such that the method can be readily reproduced by others.[1]

The remainder of the paper is organized as follows. Section II discusses related work. The regression method is described in Section III and the algorithmic details can be found in Appendix A. Section IV describes implementation details. Experimental results are presented and discussed in Section V. Finally, Section VI draws some conclusions for future work.

II. Related Work

Head-pose estimation has been very well investigated for the past decades and several powerful methods have been developed. The head-pose estimation literature was surveyed a few years ago [4], however this survey does not include methods based on depth data as these papers were published after 2009. For convenience we grouped the head-pose methods into four categories: (i) methods that are based on depth images, (ii) methods based on manifold learning, (iii) methods based on regression, and (iv) methods that combine pose with face detection and localization.

The recent advent of depth cameras enabled fast development of depth-based head-pose methods. Depth data allow to overcome some of the drawbacks of RGB data, such as illumination problems and facial landmarks detection, which is more reliable. One of the first methods that uses depth data is [14], were a depth map of the head region is combined with a color histogram and used to train a neural network. Random forest regression is proposed in [15] to estimate both head pose and facial landmarks. Random forest regression is also used in [16] where RGB SIFT descriptors are combined with 3D HOG descriptors. More recently [17] released a new method combining RGB and depth images to infer head pose. The RGB image is used to find facial landmarks, the 3D positions of the landmarks are used to fit a plane on the face that will be used to extract the 3D points that belong to the face. Using the face 3D point clouds, a morphable model of a face is map on it using optimization methods to estimate head orientation parameters. A general observation about these methods is that depth information is merely used to disambiguate photometric data and that depth data cannot be used alone for head pose.

Several authors proposed to use manifold learning, namely finding a low-dimensional output space of head poses from a high-dimensional input space of feature vectors. Nevertheless, the output variables do not necessarily correspond to the pose angles, so one has to farther learn in a supervised way the mapping between the manifold-learning output and the desired space spanned by the pose parameters. This has been achieved in various ways, e.g. [9], [18]–[23]. As already mentioned, these two step methods suffer from the fact that unsupervised manifold-learning techniques do not guarantee that the predicted output space contains the information needed for head pose.

Among the regression methods used for head pose are Gaussian process regression (GPR) [24], support vector regression (SVR) [3], partial least squares (PLS) [25] and and kernel PLS [26]. Both [24] and [3] estimate the pose angles independently, so several regression functions must be learned, one for each angle, hence correlations between these parameters cannot be taken into account. Another drawback of all kernel methods is that they require the design of a kernel function with its hyper-parameters, which must be either manually selected or properly estimated using non-convex optimization techniques.

PLS and kernel PLS proceed in two steps. First, both the input and the output are projected onto low-dimensional latent subspaces by maximizing the covariance between the projected input and the projected output. Second, a linear regression between these two latent subspaces is estimated. The performance of PLS is subject to the relationship between the covariance matrices of input and output variables and to the eigen structure of the covariance of the input variable [27]. The advantage of the proposed method is that it estimates a mixture of linear regressions directly from the input and output variables.

The methods described so far use manually annotated images for training and face detectors for testing. As already discussed, this could lead to pose estimation errors because a test feature vector may not lie in the subspace spanned by the feature vectors used for training. One way to deal with this problem is to combine face detection and pose estimation in a single task. For example, [28] considers face templates for all possible poses which are then fed into a cascaded classifier.

Convolutional neural network (CNN) architectures were also proposed in the recent past [8], [29]. [8] considers a fixed image sub-window at all locations and scales. The network consists of $64,000$ weights and kernel coefficients that need to be estimated, and both face and non-face samples must be considered. Altogether, training the network with $52,000$ positives and $52,000$ negatives involves non-linear optimization and takes 26 hours on a 2GHz Pentium 4. [29] proposed a CNN architecture composed of four convolutional layers with max-pooling on the first two layers; the activation function is the hyperbolic tangent which yields good convergence during the training phase. Small input RGB images (32×32 pixels) and small filters (5×5 pixels) are used in order to overcome the limitation of the training dataset. The network is trained using $13,500$ face patches extracted from the dataset. More recently, [30] proposed to simulate a dataset of head poses in order to train a CNN. Then they use the trained network to estimate head pose from real color images using the BIWI dataset [15]. They show that when trained using the BIWI dataset the CNN approach yields results similar to [12] and that the accuracy is improved, by a factor of 2, when a large set of simulated images are used for training.

[1]Supplementary material, that includes a complete Matlab package and examples of trained regression functions, is publicly available at https://team.inria.fr/perception/research/head-pose.

The problem of bounding-box-to-face miss-alignment was discussed and addressed in [26]. First, kernel PLS is trained on manually extracted bounding boxes and associated pose parameters, and second, head-pose prediction is applied to the bounding box found by a face detector, as well as to a number of shifted bounding boxes. The bounding box that produces the minimum PLS residual is selected for predicting the pose parameters using the trained regression. This results in a time-consuming estimator since both many bounding boxes must be considered and latent-space PLS projections must be performed at runtime. The advantage of our method with respect to [26] is that bounding box shifting is embedded in the learning process such that the optimal shift becomes part of the output, which yields a computationally efficient head pose predictor.

A short version of this paper was recently published [12]. The main addition with respect to [12] is the use of a partially-latent response variable associated with the proposed inverse regression method. This can be viewed as an augmented model and its incorporation in our formulation yields excellent results because it enables robustness to phenomena that make the head-pose estimation problem difficult. Both the proposed method (Section III) and the algorithm (Appendix A) are described in more detail than in [12], which makes the paper self-contained, self-explanatory, and enables others to easily reproduce the results. Additionally, Section V contains an extended set of results and comparisons, using three publicly available datasets.

III. Partially-Latent Mixture of Linear Regressions

In this section we summarize the generative *inverse* regression method proposed in [10]. The method is referred to as Gaussian locally linear mapping (GLLiM). GLLiM interchanges the roles of the input and output variables, such that a *low-dimensional to high-dimensional* regression problem is solved instead of a high-dimensional to low-dimensional one. The immediate consequence of using such an inverse regression model is a dramatic reduction in the number of model parameters, thus facilitating the task of training.

An additional advantage of this method is that it can be trained by adding a latent part to the output: while the high-dimensional input remains fully observed, the low-dimensional output is a concatenation of a multivariate observed variable and a multivariate latent variable. This is referred to as hybrid-GLLiM (or hGLLiM). The latent part of the output variable has a principled mathematical definition (see below) but it does not necessarily have a physical meaning. The main idea (of introducing a partially latent output) relies on the fact that variabilities present in the high-dimensional face descriptor depend on head pose, on face to bounding-box alignment, and on other phenomena (face shapes, facial expressions, gender, hair and skin color, age, etc.) that are not relevant for the task at hand. The latent part of the regression output has the interesting feature of gathering information other than pose parameters and bounding-box parameters.

A. Inverse Regression

Let $\boldsymbol{X}$, $\boldsymbol{Y}$ be two random variables, such that $\boldsymbol{X} \in \mathbb{R}^L$ denotes the low-dimensional output, e.g. pose parameters and $\boldsymbol{Y} \in \mathbb{R}^D$ ($D \gg L$) denotes the high-dimensional input, e.g. feature vector. The goal is to predict a *response* $\boldsymbol{X}$ given both an *input* $\boldsymbol{Y}$ and the model parameters $\boldsymbol{\theta}$, i.e. $p(\boldsymbol{X}|\boldsymbol{Y};\boldsymbol{\theta})$. We consider the *inverse low-to-high* mapping from the output variable $\boldsymbol{X}$ to the input variable $\boldsymbol{Y}$. This, possibly non-linear, mapping is modeled by a mixture of locally-affine transformations:

$$\boldsymbol{Y} = \sum_{k=1}^{K} \mathbb{I}(Z=k)(\mathbf{A}_k\boldsymbol{X} + \boldsymbol{b}_k + \boldsymbol{e}_k), \tag{1}$$

where $\mathbb{I}$ is the indicator function, and Z is the standard discrete latent variable: $Z = k$ if and only if $\boldsymbol{Y}$ is the image of $\boldsymbol{X}$ by the affine transformation $\mathbf{A}_k\boldsymbol{X} + \boldsymbol{b}_k$, with $\mathbf{A}_k \in \mathbb{R}^{D\times L}$ and $\boldsymbol{b}_k \in \mathbb{R}^D$, and $\boldsymbol{e}_k \in \mathbb{R}^D$ is an error vector capturing both the high-dimensional observation noise and the reconstruction error due to the piecewise approximation. The missing-data variable Z allows one to write the joint probability of $\boldsymbol{X}$ and $\boldsymbol{Y}$ as the following mixture:

$$\begin{aligned} p(\boldsymbol{Y}=\boldsymbol{y}, \boldsymbol{X}=\boldsymbol{x};\boldsymbol{\theta}) = \sum_{k=1}^{K} & p(\boldsymbol{Y}=\boldsymbol{y}|\boldsymbol{X}=\boldsymbol{x}, Z=k;\boldsymbol{\theta}) \\ & \times p(\boldsymbol{X}=\boldsymbol{x}|Z=k;\boldsymbol{\theta})p(Z=k;\boldsymbol{\theta}), \end{aligned} \tag{2}$$

where $\boldsymbol{\theta}$ denotes the model parameters and $\boldsymbol{y}$ and $\boldsymbol{x}$ denote realizations of $\boldsymbol{Y}$ and $\boldsymbol{X}$ respectively. Assuming that $\boldsymbol{e}_k$ is a zero-mean Gaussian variable with diagonal covariance matrix $\boldsymbol{\Sigma}_k \in \mathbb{R}^{D\times D}$ with diagonal entries $\sigma_{k1}, \ldots, \sigma_{kD}$, we obtain that $p(\boldsymbol{y}|\boldsymbol{x}, Z=k;\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{y};\mathbf{A}_k\boldsymbol{x}+\boldsymbol{b}_k,\boldsymbol{\Sigma}_k)$. If we further assume that $\boldsymbol{X}$ follows a mixture of Gaussians via the same assignment $Z=k$, we can write that $p(\boldsymbol{x}|Z=k;\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{x};\boldsymbol{c}_k,\boldsymbol{\Gamma}_k)$ and $p(Z=k;\boldsymbol{\theta}) = \pi_k$, where $\boldsymbol{c}_k \in \mathbb{R}^L$, $\boldsymbol{\Gamma}_k \in \mathbb{R}^{L\times L}$ and $\sum_{k=1}^{K}\pi_k = 1$. Note that this representation induces a partition of $\mathbb{R}^L$ into K regions $\mathcal{R}_k$, where $\mathcal{R}_k$ is the region where the transformation $(\mathbf{A}_k, \boldsymbol{b}_k)$ is most likely invoked, e.g. Fig. 1. This model is described by the parameter set

$$\boldsymbol{\theta} = \{\boldsymbol{c}_k, \boldsymbol{\Gamma}_k, \pi_k, \mathbf{A}_k, \boldsymbol{b}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}. \tag{3}$$

Notice that the number of model parameters $\boldsymbol{\theta}$ is dictated by the number of parameters of a multivariate Gaussian distribution and by the number of Gaussian components (K). However, the number of parameters of an unconstrained GMM is quadratic in the dimension of the variable. Hence, the size of the training dataset required to reliably estimate a conventional GMM would become prohibitively high for the dimension considered in the paper ($D = 1888$). This is why an inverse regression strategy is adopted, making the number of parameters linear in the input variable dimension rather than quadratic. This drastically reduces the model size in practice, making it tractable.

B. Inverse Regression with Partially Latent Output

We now extend the previous model such that one can train the inverse regression in the presence of partially latent

output: hybrid-GLLiM. While the high-dimensional variable $\boldsymbol{Y}$ remains unchanged, i.e. fully observed, the low-dimensional variable is a concatenation of an observed variable $\boldsymbol{T} \in \mathbb{R}^{L_t}$ and a latent variable $\boldsymbol{W} \in \mathbb{R}^{L_w}$, namely $\boldsymbol{X} = [\boldsymbol{T}; \boldsymbol{W}]$, where $[\cdot;\cdot]$ denotes vertical vector concatenation and with $L_t + L_w = L$. Hybrid-GLLiM can be seen as a latent-variable augmentation of standard regression. It can also be seen as a semi-supervised dimensionality reduction method since the unobserved low-dimensional variable $\boldsymbol{W}$ must be recovered from realizations of the observed variables $\boldsymbol{Y}$ and $\boldsymbol{T}$.

The decomposition of $\boldsymbol{X}$ implies that some of the model parameters must be decomposed as well, namely $\boldsymbol{c}_k$, $\boldsymbol{\Gamma}_k$ and $\mathbf{A}_k$. Assuming the independence of $\boldsymbol{T}$ and $\boldsymbol{W}$ given Z we have

$$\boldsymbol{c}_k = \begin{pmatrix} \boldsymbol{c}_k^t \\ \boldsymbol{c}_k^w \end{pmatrix}, \quad \boldsymbol{\Gamma}_k = \begin{pmatrix} \boldsymbol{\Gamma}_k^t & 0 \\ 0 & \boldsymbol{\Gamma}_k^w \end{pmatrix}, \quad \mathbf{A}_k = \begin{pmatrix} \mathbf{A}_k^t & \mathbf{A}_k^w \end{pmatrix}. \tag{4}$$

It follows that (1) rewrites as

$$\boldsymbol{Y} = \sum_{k=1}^{K} \mathbb{I}(Z = k)(\mathbf{A}_k^t \boldsymbol{T} + \mathbf{A}_k^w \boldsymbol{W} + \boldsymbol{b}_k + \boldsymbol{e}_k), \tag{5}$$

While the parameters to be estimated are the same, i.e. (3), there are two sets of missing variables, $Z_{1:N} = \{Z_n\}_{n=1}^N \in \{1 \dots K\}$ and $\boldsymbol{W}_{1:N} = \{\boldsymbol{W}_n\}_{n=1}^N \in \mathbb{R}^{L_w}$, associated with the training data $(\boldsymbol{y}, \boldsymbol{t})_{1:N} = \{\boldsymbol{y}_n, \boldsymbol{t}_n\}_{n=1}^N$ given the number of K of affine transformations and the latent dimension L_w. Also notice that the means $\{\boldsymbol{c}_k^w\}_{k=1}^K$ and covariances $\{\boldsymbol{\Gamma}_k^w\}_{k=1}^K$ must be fixed to avoid non-identifiability issues. Indeed, changing their values corresponds to shifting and scaling the latent variables $\{\boldsymbol{W}_n\}_{n=1}^N$ which are compensated by changes in the parameters of the affine transformations $\{\mathbf{A}_k^w\}_{k=1}^K$ and $\{\boldsymbol{b}_k^w\}_{k=1}^K$. This identifiability problem is the same as the one encountered in latent variable models for dimension reduction and is always solved by fixing these parameters. Following [31] and [32], the means and covariances are fixed to zero and to the identity matrix respectively: $\boldsymbol{c}_k^w = \mathbf{0}, \boldsymbol{\Gamma}_k^w = \mathbf{I}, \forall k \in \{1 \dots K\}$.

The corresponding EM algorithm consists of estimating the parameter set $\boldsymbol{\theta}$ that maximizes

$$\boldsymbol{\theta}^{(i)} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \left(\mathbb{E}_Z [\log p((\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}, Z)_{1:N}; \boldsymbol{\theta}| (\boldsymbol{x}, \boldsymbol{y})_{1:N}; \boldsymbol{\theta}^{(i-1)})] \right). \tag{6}$$

Using that $\boldsymbol{W}_{1:N}$ and $\boldsymbol{T}_{1:N}$ are independent conditionally on $Z_{1:N}$ and that $\{\boldsymbol{c}_k^{\mathrm{w}}\}_{k=1}^K$ and $\{\boldsymbol{\Gamma}_k^{\mathrm{w}}\}_{k=1}^K$ are fixed, maximizing (6) is then equivalent to

$$\boldsymbol{\theta}^{(i)} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \left\{ \mathbb{E}_{r_Z^{(i)}} [\mathbb{E}_{r_{W|Z}^{(i)}} [\log p(\boldsymbol{y}_{1:N} \mid (\boldsymbol{t}, \mathbf{W}, Z)_{1:N}; \boldsymbol{\theta})] + \log p((\boldsymbol{t}, Z)_{1:N}; \boldsymbol{\theta})] \right\}, \tag{7}$$

where $r_Z^{(i)}$ and $r_{W|Z}^{(i)}$ denote the posterior distributions

$$r_{W|Z}^{(i)} = p(\mathbf{W}_{1:N} | (\boldsymbol{y}, \boldsymbol{t}, Z)_{1:N}; \boldsymbol{\theta}^{(i-1)}), \tag{8}$$

$$r_Z^{(i)} = p(\mathbf{Z}_{1:N} | (\boldsymbol{y}, \boldsymbol{t})_{1:N}; \boldsymbol{\theta}^{(i-1)}). \tag{9}$$

It follows that the E-step splits into two steps, an E-W step and an E-Z step. Details of the hybrid-GLLiM algorithm are provided in Appendix A. Note that the model described in Section III-A simply corresponds to $L_w = 0$, hence this algorithm can be used to solve both models (fully observed output and partially observed output).

C. Forward Predictive Distribution

Once the model parameters $\boldsymbol{\theta}$ are estimated, one obtains the low-dimensional to high-dimensional *inverse predictive distribution* as detailed in [10]. More interesting, it is also possible to obtain the desired high-dimensional to low-dimensional *forward predictive distribution*:

$$p(\boldsymbol{x}|\boldsymbol{y}; \boldsymbol{\theta}^*) = \sum_{k=1}^{K} \nu_k^* \mathcal{N}(\boldsymbol{x}; \mathbf{A}_k^* \boldsymbol{y} + \boldsymbol{b}_k^*, \boldsymbol{\Sigma}_k^*) \tag{10}$$

$$\text{with } \nu_k^* = \frac{\pi_k^* \mathcal{N}(\boldsymbol{y}; \boldsymbol{c}_k^*, \boldsymbol{\Gamma}_k^*)}{\sum_{j=1}^{K} \pi_j^* \mathcal{N}(\boldsymbol{y}; \boldsymbol{c}_j^*, \boldsymbol{\Gamma}_j^*)} \tag{11}$$

which is also a Gaussian mixture conditioned by the parameters $\boldsymbol{\theta}^*$:

$$\boldsymbol{\theta}^* = \{\boldsymbol{c}_k^*, \boldsymbol{\Gamma}_k^*, \pi_k^*, \mathbf{A}_k^*, \boldsymbol{b}_k^*, \boldsymbol{\Sigma}_k^*\}_{k=1}^K. \tag{12}$$

A prominent feature of this model is that the parameters $\boldsymbol{\theta}^*$ can be expressed analytically from the parameters $\boldsymbol{\theta}$ (please consult the analytical expressions of these parameters in Appendix A):

$$\boldsymbol{c}_k^* = \mathbf{A}_k \boldsymbol{c}_k + \boldsymbol{b}_k, \tag{13}$$

$$\boldsymbol{\Gamma}_k^* = \boldsymbol{\Sigma}_k + \mathbf{A}_k \boldsymbol{\Gamma}_k \mathbf{A}_k^\top, \tag{14}$$

$$\pi_k^* = \pi_k, \tag{15}$$

$$\mathbf{A}_k^* = \boldsymbol{\Sigma}_k^* \mathbf{A}_k^\top \boldsymbol{\Sigma}_k^{-1}, \tag{16}$$

$$\boldsymbol{b}_k^* = \boldsymbol{\Sigma}_k^* \left(\boldsymbol{\Gamma}_k^{-1} \boldsymbol{c}_k - \mathbf{A}_k^\top \boldsymbol{\Sigma}_k^{-1} \boldsymbol{b}_k \right), \tag{17}$$

$$\boldsymbol{\Sigma}_k^* = \left(\boldsymbol{\Gamma}_k^{-1} + \mathbf{A}_k^\top \boldsymbol{\Sigma}_k^{-1} \mathbf{A}_k \right)^{-1}. \tag{18}$$

Using (10) one can predict an output $\hat{\boldsymbol{x}}$ corresponding to a test input $\hat{\boldsymbol{y}}$

$$\hat{\boldsymbol{x}} = f(\hat{\boldsymbol{y}}) \quad \text{with:}$$

$$f(\boldsymbol{y}) = \mathbb{E}[\boldsymbol{x}|\boldsymbol{y}; \boldsymbol{\theta}^*] = \sum_{k=1}^{K} \nu_k^* \left(\mathbf{A}_k^* \boldsymbol{y} + \boldsymbol{b}_k^* \right). \tag{19}$$

IV. Implementation Details

The proposed head-pose estimation method is implemented as follows. We use the Matlab computer vision toolbox implementation of the face detector of [33] as we found that this method yields very good face detections and localizations for a wide range of face orientations, including side views. The Matlab implementation of [33] offers three different trained classifiers for face detection: two of them for frontal-view detection and one for profile-view detection. These three classifiers yield different results for face detection in terms of bounding-box location and size. The results of face detection using these three classifiers are then combined together for both training and testing of our method. For

each face detection the associated bounding box is resized to patches of 64×64, converted to a grey-level image to which histogram equalization is then applied. A HOG descriptor is extracted from this resized and histogram-equalized patch. To do so, we build a HOG pyramid (p-HOG) by stacking HOG descriptors at multiple resolutions. The following parameters are used to build p-HOG descriptors:

- Block resolution: 2×2 cells;
- Cell resolutions: 32×32, 16×16 and 8×8, and
- Number of orientation bins: 8

Three HOG descriptors are computed, one for each cell resolution, which are then stacked to form a high-dimensional vector $\boldsymbol{y} \in \mathbb{R}^D$, with $D = 1888$.

The dimension of the output variable $\boldsymbol{x} \in \mathbb{R}^L$ depends on the number of pose parameters (up to three angles: yaw, pitch and roll), the bounding-box shift parameters (horizontal and vertical shifts) and the number of latent variables. Hence the output dimension may vary from $L = 1$ (one angle, no shift, no latent variable) to $L = 10$ (three angles, two shifts, four latent variables).

We used the algorithm detailed in Appendix A to learn the model parameters, either with fully observed output variables (pose angles and bounding-box shifts), or with both observed and latent output variables. From these inverse regression parameters, we derive the forward parameters $\boldsymbol{\theta}^*$, i.e. equations (13)-(18) that allow us to estimate the forward predictive distribution (10), and hence to predict an output from a test input, i.e. (19). The joint estimation of the head-pose angles and of the bounding-box shift is achieved iteratively in the following way. The current bounding-box location, $\boldsymbol{u} \in \mathbb{R}^2$, is used to build a feature vector $\boldsymbol{y}$ from which both a head pose $\boldsymbol{x}_h$ and a bounding-box shift $\boldsymbol{x}_b$ are predicted. The latter is then used to update the bounding-box location, to build an updated feature vector and to predict an updated head pose and a new bounding-box shift. This iterative prediction is described in detail in Algorithm 1.

Algorithm 1 Iterative prediction

Require: Bounding-box location $\boldsymbol{u}$ and forward model parameters $\boldsymbol{\theta}^*$

procedure HEADPOSEESTIMATION($\boldsymbol{u}$,$\boldsymbol{\theta}^*$)
  repeat
    Build $\boldsymbol{y}$ from current bounding-box location $\boldsymbol{u}$
    Predict $\boldsymbol{x} = [\boldsymbol{x}_h; \boldsymbol{x}_b]$ from $\boldsymbol{y}$ using (19)
    Update the bounding-box location $\boldsymbol{u} = \boldsymbol{u} + \boldsymbol{x}_b$
  until $\|\boldsymbol{x}_b\| \leq \epsilon$
  return head-pose $\boldsymbol{x}_h$ and bounding-box location $\boldsymbol{x}_b$
end procedure

V. EXPERIMENTS

In this section we present an experimental evaluation of the proposed head-pose estimation methodology. The experiments are carried out with three publicly available datasets: the Prima dataset [34], the BIWI dataset [15], and the McGill real-world face video dataset [35], [36]:

- The **Prima head pose dataset** consists of 2790 images of 15 persons recorded twice. Pitch values lie in the interval $[-60°, 60°]$, and yaw values lie in the interval $[-90°, 90°]$ with a $15°$ step. Thus, there are 93 poses available for each person. All the recordings were achieved with the same background. One interesting feature of this dataset is the the pose space is uniformly sampled. The dataset is annotated such that a face bounding box (manually annotated) and the corresponding yaw and pitch angle values are provided for each sample.
- The **Biwi Kinect head pose dataset** consists of video recordings of 20 people (16 men, 4 women, some of them recorded twice) using a Kinect camera. During the recordings, the participants freely move their head and the corresponding head angles lie in the intervals $[-60°, 60°]$ (pitch), $[-75°, 75°]$ (yaw), and $[-20°, 20°]$ (roll). Unlike the Prima dataset, the parameter space is not evenly sampled. The face centers (nose tips) were detected on each frame in the dataset, which allowed us to automatically extract a bounding box for each sample.
- The **McGill real-world face video dataset** consists of 60 videos (a single participant per video, 31 women and 29 men) recorded with the objective of studying unconstrained face classification. The videos were recorded in different environments (both indoor or outdoor) thus resulting in arbitrary illumination conditions and background clutter. Furthermore, the participants were completely free in their behaviors and movements. Yaw angles range in the interval $[-90°, 90°]$. Yaw values corresponding to each video frame are estimated using a two-step labeling procedure that provides the most likely angle as well as a degree of confidence. The labeling consists of showing images and possible angle values to human experts, i.e. [35].

We adopted the leave-one-out evaluation protocol at the individual person level. More precisely, all the images/frames associated with one participant are left aside and used for testing, while the remaining data were used for training. As a measure of performance, we evaluated the absolute error between an estimated angle and the ground-truth value, then we computed the *mean absolute error* (MAE) and standard deviation (STD) over several tests.

We experimented with the following variants of the proposed algorithm (notice that the number of pose angles depends on the dataset, i.e. yaw, pitch and roll (BIWI), yaw and pitch (Prima) or yaw (McGill)):

- *GLLiM_pose* learns and predicts with one, two, or three pose angles;
- *hGLLiM_pose-d* learns with the pose parameters as well as with partially latent output, where the number d of latent variables varies between 1 and 4;
- *GLLiM_pose&bb* learns and predicts with both pose angles and bounding-box shifts, and

- *hGLLiM_pose&bb-d* learns and predicts with pose angles, bounding-box shifts and partially latent output.

An important aspect of any head-pose method is the way faces are detected in images. We used manually annotated bounding boxes, whenever they are available with the datasets. Otherwise, we used bounding boxes provided with a face detector, e.g. [33]. To evaluate the robustness to inaccurate face localization, we introduced random shifts, drawn from a Gaussian distribution, and we used these shifts in conjunction with *GLLiM_pose&bb* and with *hGLLiM_pose&bb-d* to learn the regression parameters and to predict the correct bounding-box location. Notice that in the case of the latter algorithms, the prediction is run iteratively, i.e. Algorithm 1: Extract a HOG vector, predict the pose and the shift, extract a HOG vector from the shifted bounding box, predict the pose and the shift, etc. This is stopped when the shift becomes very small and, as it can be seen below, it considerably improves the quality of the head-pose method.

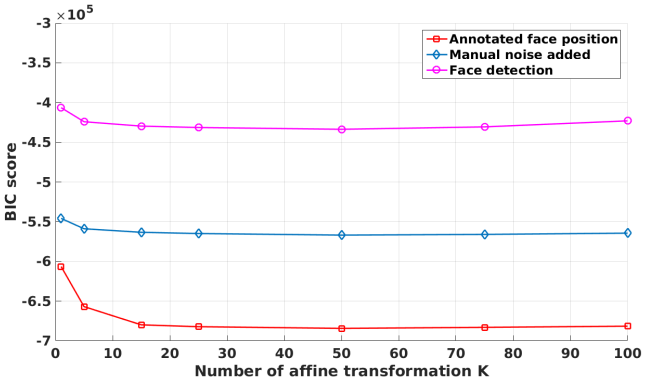

Fig. 2. The Bayesian information criterion (BIC) as a function of the number of affine transformations in GLLiM. These experiments use the Prima dataset with the leave-one-out protocol.

TABLE I
THE BIC SCORE FOR SEVERAL MODELS LEARNED WITH DIFFERENT VALUES OF K USING THE PRIMA DATASET. WE USE GLLIM_POSE TO LEARN EACH MODEL WITH DIFFERENT INPUT DATA (FIG. 2): ANNOTATED FACE POSITION (AFP), ADDING MANUAL NOISE TO THE FACE POSITION (MNA) AND USING A FACE DETECTOR (FD), THE OPTIMAL BIC SCORES ARE IN BOLD.

Data	$K=1$	$K=5$	$K=25$	$K=50$	$K=100$
AFP	-6.0608	-6.5845	-6.822	$\mathbf{-6.8429}$	-6.8173
MNA	-5.4554	-5.6018	-5.6491	$\mathbf{-5.6688}$	-5.6455
FD	-4.0596	-4.2602	-4.3144	$\mathbf{-4.3366}$	-4.2307

The number K of Gaussian components is an important parameter, which in our model corresponds to the number of affine mappings. We carried out several experiments in order to evaluate the quality of the results obtained by our method as a function of the number of affine transformations in the mixture. We use the GLLiM_pose variant of our algorithm with three different face detection options: manual annotation, manual annotation perturbed with additive Gaussian noise, and automatic face detection. We trained these three versions of GLLiM_pose with K varying from 1 to 100.

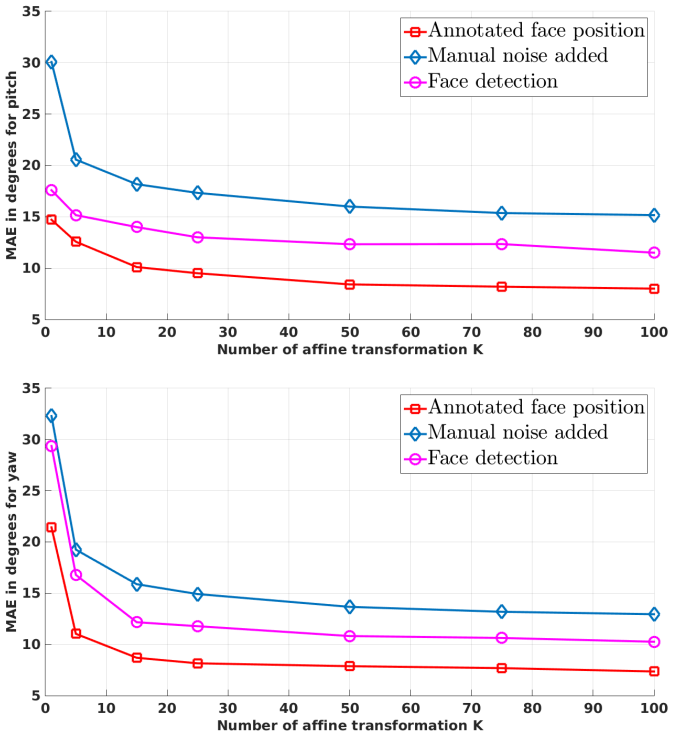

Fig. 3. Mean absolute error (MAE) in degrees, for pitch (top) and yaw (bottom), as a function of the number of affine transformations in the mixture of linear regression model. We used GLLiM_pose to learn the model parameters independently for pitch and yaw. The three curves correspond to the following face detection cases: manual annotation (red curve), manual annotations with additive noise (blue), and automatic face detection (magenta). These experiments use the Prima dataset with the leave-one-out protocol.

In order to determine the optimal number of affine mappings, K, associated with GLLiM, we use two criteria, the Bayesian information criterion (BIC) which is an information-theoretic criterion generally used for model selection, and an experimental figure of merit based on the mean absolute error (MAE). We learned several models for different values of K using the Prima dataset. We seek the model that yields low BIC and MAE scores. The BIC and MAE values are plotted as a function of K in Table I and in Fig. 2 and Fig. 3. These curves show the same behavior: as the number of affine mappings increases from $K=1$ to approximatively $K=30$, both the BIC and MAE scores decrease, then the curve slopes become almost horizontal. Both BIC and MAE reach the lowest score for $K=50$. This behavior can be explained as follows. When $K<5$ the model is not flexible enough to take into account the apparently non-linear mapping between HOG features and head-pose parameters. It can be observed from Fig. 3 that a large value for K increases the model accuracy. As expected, the computational complexity increases with K as well. Indeed, the number of model parameters is linear in the number of mixture components and hence the size of the training dataset must be increased as well. It is well known that a large number of components in a mixture model presents the risk of overfitting. It is interesting to notice that BIC (derived

TABLE II
MEAN ABSOLUTE ERROR (MAE) AND STANDARD DEVIATION (STD) (IN DEGREES) OBTAINED WITH VARIOUS HEAD-POSE METHODS, REGRESSION METHODS, AND OUR METHOD USING THE PRIMA DATASET. THIS DATASET CONTAINS MANUALLY ANNOTATED BOUNDING BOXES OF FACES AND THE CORRESPONDING PITCH AND YAW ANGLES. IN ORDER TO TEST THE ROBUSTNESS WE SIMULATED SHIFTED BOUNDING BOXES. THE BEST RESULTS ARE IN BOLD.

	Manually annotated bounding boxes				Manual annotation + simulated shifts			
	Pitch		Yaw		Pitch		Yaw	
Method	MAE	STD	MAE	STD	MAE	STD	MAE	STD
Stiefelhagen [37]‡	9.7	-	9.5	-	-	-	-	-
Gourier et al. [38]‡	12.1	-	7.3	-	-	-	-	-
GPR [39]†	11.94	10.19	15.04	12.24	19.96	16.58	23.69	18.16
PLS [40]†	12.25	9.73	13.38	10.8	17.77	14.47	17.34	13.94
SVR [41]†	11.25	9.42	12.82	10.99	17.09	14.81	17.27	14.09
GLLiM_pose	8.41	10.65	7.87	8.08	15.99	16.69	13.66	14.78
hGLLiM_pose-2	**8.47**	**10.35**	**7.93**	**7.9**	12.64	14.49	11.51	11.37
hGLLiM_pose-4	8.5	10.8	7.85	7.98	**12.03**	13.92	**10.78**	9.77
GLLiM_pose&bb	-	-	-	-	13.13	13.65	11.3	10.55
hGLLiM_pose&bb-2	-	-	-	-	12.52	**12.44**	11.04	9.7
hGLLiM_pose&bb-4	-	-	-	-	12.12	12.85	11.27	**9.53**

TABLE III
THE MEAN ABSOLUTE ERROR (MAE) AND STANDARD DEVIATION (STD), EXPRESSED IN DEGREES, OBTAINED WITH VARIOUS HEAD-POSE METHODS, REGRESSION METHODS, AND OUR METHOD USING THE BIWI DATASET. THIS DATASET CONTAINS ANNOTATED BOUNDING BOXES OF FACES AND THE CORRESPONDING PITCH, YAW, AND ROLL ANGLE VALUES. IN ORDER TO TEST THE ROBUSTNESS WE SIMULATED SHIFTED BOUNDING BOXES BOTH FOR TRAINING AND FOR TESTING. THE BEST RESULTS ARE IN BOLD. NOTE THAT [15] USES DEPTH DATA ONLY AND [42] AND [17] USE BOTH COLOR AND DEPTH INFORMATION. PAPERS USING DEPTH DATA ARE MARKED WITH A *.

	Manually annotated bounding boxes						Manual annotation + simulated shifts					
	Pitch		Yaw		Roll		Pitch		Yaw		Roll	
Method	MAE	STD	MAE	STD	MAE	STD	MAE	STD	MAE	STD	MAE	STD
Ghiass et al. [17]‡*	**0.1**	6.7	**0.2**	8.7	**0.3**	9.3	-	-	-	-	-	-
Fanelli et al. [15]‡*	3.8	6.5	3.5	6.8	5.4	6.0	-	-	-	-	-	-
Wang et al. [42]‡*	8.5	11.1	8.8	14.3	7.4	10.8	-	-	-	-	-	-
GPR [39]†	9.64	8.85	7.72	7.17	6.01	6.29	10.77	9.45	9.06	8.33	6.54	6.72
PLS [40]†	7.87	6.73	7.35	6.06	6.11	5.9	10.32	8.64	8.67	7.7	6.69	6.74
SVR [41]†	7.77	6.85	6.98	6.26	5.14	5.96	10.82	9.22	9.14	8.32	6.26	7.16
GLLiM_pose	5.77	5.77	4.48	**4.33**	4.71	5.31	11.33	11.58	10.2	11.34	7.76	8.02
hGLLiM_pose-2	5.57	5.48	4.33	4.68	4.37	5.09	9.04	9.13	7.65	8.3	6.3	6.75
hGLLiM_pose-4	5.43	**5.44**	4.24	5.37	4.13	**4.86**	8.45	8.41	6.93	7.72	6.12	6.8
GLLiM_pose&bb	-	-	-	-	-	-	8.49	8.79	6.86	7.3	6.57	6.95
hGLLiM_pose&bb-2	-	-	-	-	-	-	7.81	7.68	6.41	7.19	5.75	6.68
hGLLiM_pose&bb-4	-	-	-	-	-	-	**7.65**	**8.0**	**6.06**	**6.91**	**5.62**	**6.35**

from information theory) and MAE (based on experiments with the data) yield the same optimal value, namely $K \approx 50$.

The proposed algorithms were compared with the following state-of-the-art head-pose estimation methods: the neural-network based methods of [37], [38] and of [29] , the method of [43] based on dictionary learning, the graphical-model method of [44], the template based method of [28], the supervised non-linear optimization method of [45], the optimization method of [17], and the random-forest methods of [15] and of [42]. Additionally, we benchmarked the following regression methods: support vector regression (SVR) [41], Gaussian process regression (GPR) [39], and partial least squares (PLS) [40], as they are widely known and commonly used regression methods for which software packages are publicly available. Notice that some of these methods estimate only one parameter, i.e. the yaw angle [28], [43]–[45], while the random-forest methods of [15], [17] and [42] use depth information available with the BIWI (Kinect) dataset.

Table II, Table III, and Table IV show the results of head-pose estimation obtained with the Prima, BIWI, and McGill datasets, respectively. The † symbol indicates that the results are those reported by the authors while the ‡ symbol indicates that the results are obtained using either publicly available software packages or our own implementations. In the case of the Prima dataset, GLLiM_pose and hGLLiM_pose yield the best results. We note that hGLLiM_pose&bb variants of the algorithm (simultaneous prediction of pose, bounding-box shift and partially-latent output) increase the confidence (low STD). Table III shows the results obtained with the BIWI datasets. As already mentioned, [15] uses depth information and [42], [17] use of depth and color information. Overall, the proposed algorithms compare favorably with [15]. hGLLiM_pose-4 yields the best MAE for the roll angle, while [17] yields the best MAE for pitch and yaw, but with a high standard deviation. Our algorithm estimates the parameters with the highest confidence (lowest standard deviation). Table IV shows the results obtained with the McGill dataset. The ground-truth yaw values in this dataset are obtained by human experts that must choose among a discrete set of 7 values. Clearly, this is not enough to properly train our algorithms. The method of [44] yields the best results in terms of RMSE while hGLLiM_pose-2 yields the best results in terms of MAE. Notice that PLS yield the highest confidence in this case.

The results summarized in Table II, Table III and Table IV allow to quantify the variants of the proposed algorithm. With manually annotated bounding boxes, e.g. Tables II and III, there is no notable difference between these variants. Whenever the regression is trained with simulated bounding-box shifts, both GLLiM_pose&bb and hGLLiM_pose perform better than GLLiM_pose. It is interesting to note that in the case of the Prima dataset (Table II), hGLLiM_pose-4 (the dimension of the latent part of the output is 4) is the best-performing variant of GLLiM, while in the case of the BIWI dataset (Table III), hGLLiM_Pose&bb-4 is the best performing one. We also experimented with a larger latent dimension without improving the accuracy. Concerning the McGill dataset, neither GLLiM_pose&bb nor hGLLiM_pose&bb could be

TABLE IV
ROOT MEAN SQUARE ERROR (RMSE), MEAN ABSOLUTE ERROR (MAE) AND STANDARD DEVIATION (STD) (IN DEGREES) OBTAINED WITH VARIOUS HEAD-POSE METHODS, REGRESSION METHODS, AND OUR METHOD USING THE MCGILL REAL-WORLD DATASET. THIS DATASET CONTAINS ANNOTATED YAW ANGLES. BOUNDING BOXES ARE LOCATED WITH A FACE DETECTOR. THE BEST RESULTS ARE IN BOLD.

	Bounding boxes based on face detection		
	Yaw		
Method	RMSE	MAE	STD
Demirkus et al. [43]‡	> 40	-	-
Xiong and De la Torre [45]‡	29.81	-	-
Zhu and Ramanan [28]‡	35.70	-	-
Demirkus et al. [44]‡	**12.41**	-	-
GPR [39]†	23.18	16.22	16.71
PLS [40]†	22.46	15.56	**16.2**
SVR [41]†	21.13	15.25	18.43
GLLiM_pose	26.62	13.1	23.17
hGLLiM_pose-2	24.0	**11.99**	20.79
hGLLiM_pose-4	24.25	12.01	21.06

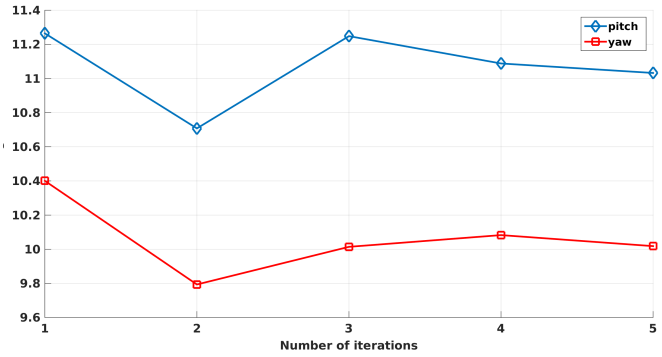

Fig. 4. Mean absolute error (MAE) for pitch and yaw as a function of the number of iterations of GLLiM_pose&bb.

trained because the ground-truth face bounding boxes are not available. In the case of this dataset, we used a face detector both for training and for testing. The fact that, overall, hGLLiM_pose performs better than GLLiM_pose validates the advantage of adding a latent component to the output variable. The latter "absorbs" various phenomena that would otherwise affect the accuracy of the pose parameters.

As already mentioned, GLLiM_pose&bb and hGLLiM_pose&bb are applied iteratively, until there is no improvement in the predicted output. Fig. 4 plots the MAE of yaw and pitch as a function of the number of iterations of GLLiM_pose&bb. As it can be observed from these curves, the MAE decreases after two iterations and then it slightly increases and again it decreases. Therefore, in

Fig. 5. Examples of simultaneous estimation of head-pose angles and of bounding-box shifts. The initial bounding box (found with an automatic face detector) is shown in red. The estimated bounding box is shown in green.

practice we run two iterations of these algorithms.

Finally, we applied hGLLiM_Pose&bb-4 to a set of images that are not contained in any of the three datasets. Fig. 5 shows some of these results. The face detector of [33] was used to detect faces (red bounding boxes). The output of hGLLiM_Pose&bb-4: the shifted bounding boxes (shown in green) and the estimated pitch, roll, and yaw angles (shown with a green arrow).

VI. Conclusions

In this paper we proposed a solution to the problem of estimating head pose from the bounding box aligned with a face. Instead of extracting facial landmarks, the method directly maps high-dimensional feature vectors (extracted from faces) onto a low-dimensional manifold. The method relies on learning a mixture of linear regressions. The latter is modeled within the framework of generative methods. More precisely, it is assumed that the high-dimensional feature space is generated from a low-dimensional parameter space. Consequently, an inverse regression strategy is adopted: a low-dimensional to high-dimensional regression is learnt, followed by Bayes inversion.

We experimented with four variants of the proposed algorithm: (i) GLLiM_pose, which learns and predicts the pose parameters, (ii) hGLLiM_pose, which learns and predicts the pose parameters in the presence of latent variable augmentation of the output, (iii) GLLiM_pose&bb, which learns and predicts both the pose parameters and bounding-box shifts, and (iv) hGLLiM_pose&bb which combines hGLLiM with GLLiM_pose&bb. The experiments and benchmarks, carried out with three publicly available datasets, show that the latent-augmentation variants of the algorithm improve the accuracy of the estimation and perform better than several state-of-the-art algorithms.

The methodology presented in this paper has not been tuned for the particular application of head-pose estimation. The algorithms may be applied, with minor modifications, to other high-dimensional to low-dimensional mapping problems, e.g. estimation of human pose from color and depth images. It may also be used as input for gaze estimation or for determining the visual focus of attention, e.g. [46].

In the future we plan to extend the method to the problem of tracking pose parameters. Indeed, a natural extension of the proposed method is to incorporate a dynamic model to better predict the output variables over time. Such a model could be simultaneously applied to the pose parameters and to the bounding-box shifts. Hence, one can track the image region of interest and the pose parameters in a unified framework.

APPENDIX A
EM FOR GLLIM AND HYBRID-GLLIM

This appendix details the EM algorithm that estimates the parameters of the regression method described in Section III. The interest reader is referred to [10] for an in-depth description and discussion. Once initialized, at each iteration i, the algorithm alternates between two expectation steps, E-Z and E-W, and two maximization steps, M-GMM and M-mapping:

- **E-W-step:** Given the current parameter estimates $\boldsymbol{\theta}^{(i-1)}$, the posterior probability $r_{W|Z}^{(i)}$ in (8) is fully determined by the distributions $p(\boldsymbol{w}_n|Z_n=k,\boldsymbol{t}_n,\boldsymbol{y}_n;\boldsymbol{\theta}^{(i-1)})$ for all n and k, which can be shown to be Gaussian. Their covariance matrices $\mathbf{S}_k^{w(i)}$ and vector means $\boldsymbol{\mu}_{nk}^{w(i)}$ are given by

$$\mathbf{S}_k^{w(i)} = \left(\mathbf{I} + \mathbf{A}_k^{w(i-1)\top}\boldsymbol{\Sigma}_k^{(i-1)^{-1}}\mathbf{A}_k^{w(i-1)}\right)^{-1}, \tag{20}$$

$$\boldsymbol{\mu}_{nk}^{w(i)} = \mathbf{S}_k^{w(i)}\mathbf{A}_k^{w(i-1)\top}\boldsymbol{\Sigma}_k^{(i-1)^{-1}} \times \left(\boldsymbol{y}_n - \mathbf{A}_k^{t(i-1)}\boldsymbol{t}_n - \boldsymbol{b}_k^{(i-1)}\right). \tag{21}$$

- **E-Z-step:** The posterior probability $r_Z^{(i)}$ in (9) is determined by

$$r_{nk}^{(i)} = p(Z_n=k|\boldsymbol{t}_n,\boldsymbol{y}_n;\boldsymbol{\theta}^{(i-1)}) = = \frac{\pi_k^{(i-1)}p(\boldsymbol{y}_n,\boldsymbol{t}_n|Z_n=k;\boldsymbol{\theta}^{(i-1)})}{\sum_{j=1}^K \pi_j^{(i-1)}p(\boldsymbol{y}_n,\boldsymbol{t}_n|Z_n=j;\boldsymbol{\theta}^{(i-1)})} \tag{22}$$

for all n and k, where

$$p(\boldsymbol{y}_n,\boldsymbol{t}_n|Z_n=k;\boldsymbol{\theta}^{(i-1)}) = \mathcal{N}(\boldsymbol{t}_n;\boldsymbol{c}_k^t,\boldsymbol{\Gamma}_k^t)\mathcal{N}(\boldsymbol{y}_n;\boldsymbol{d}_k,\boldsymbol{\Phi}_k), \tag{23}$$

with:

$$\boldsymbol{d}_k = \mathbf{A}_k^{t(i-1)}\boldsymbol{t}_n + \boldsymbol{b}_k^{(i-1)},$$
$$\boldsymbol{\Phi}_k = \mathbf{A}_k^{w(i-1)}\mathbf{A}_k^{w(i-1)\top} + \boldsymbol{\Sigma}_k^{(i-1)}.$$

The maximization (7) can then be performed using the posterior probabilities $r_{nk}^{(i)}$ and the sufficient statistics $\boldsymbol{\mu}_{nk}^{w(i)}$ and $\mathbf{S}_k^{w(i)}$. We use the following notations: $\rho_{nk}^{(i)} = r_{nk}^{(i)}/\sum_{n=1}^N r_{nk}^{(i)}$ and $\boldsymbol{x}_{nk}^{(i)} = [\boldsymbol{t}_n;\boldsymbol{\mu}_{nk}^{w(i)}] \in \mathbb{R}^L$. The M-step can be divided into two separate steps.

- **M-GMM-step:** The updating of parameters $\pi_k^{(i)}, \boldsymbol{c}_k^{t(i)}$ and $\boldsymbol{\Gamma}_k^{t(i)}$ correspond to those of a standard Gaussian mixture model on $\boldsymbol{T}_{1:N}$, so that we get straightforwardly:

$$\boldsymbol{c}_k^{t(i)} = \sum_{n=1}^N \rho_{nk}^{(i)}\boldsymbol{t}_n, \tag{24}$$

$$\boldsymbol{\Gamma}_k^{t(i)} = \sum_{n=1}^N \rho_{nk}^{(i)}(\boldsymbol{t}_n - \boldsymbol{c}_k^{t(i)})(\boldsymbol{t}_n - \boldsymbol{c}_k^{t(i)})^\top \tag{25}$$

$$\pi_k^{(i)} = \frac{\sum_{n=1}^N r_{nk}^{(i)}}{N}. \tag{26}$$

- **M-mapping-step:** The updating of mapping parameters $\{\mathbf{A}_k,\boldsymbol{b}_k,\boldsymbol{\Sigma}_k\}_{k=1}^K$ is also in closed-form. The affine transformation matrix is updated with:

$$\mathbf{A}_k^{(i)} = \mathbf{Y}_k^{(i)}\mathbf{X}_k^{(i)\top}(\mathbf{S}_k^{\mathrm{x}(i)} + \mathbf{X}_k^{(i)}\mathbf{X}_k^{(i)\top})^{-1} \tag{27}$$

where:

$$\mathbf{X}_k^{(i)} = \left(\sqrt{\rho_{1k}^{(i)}}(\boldsymbol{x}_{1k}^{(i)} - \boldsymbol{x}_k^{(i)}), \ldots, \sqrt{\rho_{nk}^{(i)}}(\boldsymbol{x}_{Nk}^{(i)} - \boldsymbol{x}_k^{(i)})\right),$$
$$\mathbf{Y}_k^{(i)} = \left(\sqrt{\rho_{1k}^{(i)}}(\boldsymbol{y}_1 - \boldsymbol{y}_k^{(i)}), \ldots, \sqrt{\rho_{nk}^{(i)}}(\boldsymbol{y}_N - \boldsymbol{y}_k^{(i)})\right),$$
$$\boldsymbol{x}_k^{(i)} = \sum_{n=1}^N \rho_{nk}^{(i)}\boldsymbol{x}_{nk}^{(i)},$$
$$\boldsymbol{y}_k^{(i)} = \sum_{n=1}^N \rho_{nk}^{(i)}\boldsymbol{y}_n,$$
$$\mathbf{S}_k^{\mathrm{x}(i)} = \begin{pmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{S}_k^{w(i)}\end{pmatrix}.$$

The intercept parameters are updated with:

$$\boldsymbol{b}_k^{(i)} = \sum_{n=1}^N \rho_{nk}^{(i)}(\boldsymbol{y}_n - \mathbf{A}_k^{(i)}\boldsymbol{x}_{nk}^{(i)}). \tag{28}$$

The noise covariance matrices are updated with:

$$\boldsymbol{\Sigma}_k^{(i)} = \mathrm{diag}\Big\{\mathbf{A}_k^{w(i)}\mathbf{S}_k^{w(i)}\mathbf{A}_k^{w(i)\top} + \sum_{n=1}^N \rho_{nk}^{(i)}(\boldsymbol{y}_n - \mathbf{A}_k^{(i)}\boldsymbol{x}_{nk}^{(i)} - \boldsymbol{b}_k^{(i)})(\boldsymbol{y}_n - \mathbf{A}_k^{(i)}\boldsymbol{x}_{nk}^{(i)} - \boldsymbol{b}_k^{(i)})^\top\Big\} \tag{29}$$

where the $\mathrm{diag}\{\cdot\}$ operator sets all the off-diagonal entries to 0.

- **Initialization:** Initial parameters $\boldsymbol{\theta}^{(0)}$ are obtained by fitting a GMM with K components to the joint output-input training dataset $\{\boldsymbol{t}_n,\boldsymbol{y}_n\}_{n=1}^N$.

Note that the following derivations are also valid for the estimation of the parameter set $\boldsymbol{\theta}$ in (3), which corresponds to $L_w = 0$, hence the E-W step disappears.

REFERENCES

[1] S. Sabanovic, M. Michalowski, and R. Simmons, "Robots in the wild: observing human-robot social interaction outside the lab," in *IEEE International Workshop on Advanced Motion Control*, 2006, pp. 596–601.

[2] M. A. Goodrich and A. C. Schultz, "Human-robot interaction: A survey," *Foundation Trends Human-Computer Interaction*, vol. 1, no. 3, pp. 203–275, Feb 2007.

[3] E. Murphy-Chutorian, A. Doshi, and M. Trivedi, "Head pose estimation for driver assistance systems: A robust algorithm and experimental evaluation," in *IEEE Intelligent Transportation Systems Conference*, Sept 2007, pp. 709–714.

[4] E. Murphy-Chutorian and M. M. Trivedi, "Head pose estimation in computer vision: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 4, pp. 607–626, April 2009.

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[6] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, June 2005, pp. 886–893 vol. 1.

[7] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[8] M. Osadchy, Y. LeCun, and M. L. Miller, "Synergistic face detection and pose estimation with energy-based models," *Journal of Machine Learning Research*, vol. 8, pp. 1197–1215, 2007.

[9] J. Foytik and V. Asari, "A two-layer framework for piecewise linear manifold-based head pose estimation," *International Journal of Computer Vision*, vol. 101, no. 2, pp. 270–287, January 2013.

[10] A. Deleforge, F. Forbes, and R. Horaud, "High-dimensional regression with gaussian mixtures and partially-latent response variables," *Statistics and Computing*, vol. 25, no. 5, pp. 893–911, 2015.

[11] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[12] V. Drouard, S. Ba, G. Evangelidis, A. Deleforge, and R. Horaud, "Head pose estimation via probabilistic high-dimensional regression," in *IEEE International Conference on Image Processing*, September 2015, pp. 4624–4628.

[13] K.-C. Li, "Sliced inverse regression for dimension reduction," *Journal of the American Statistical Association*, vol. 86, no. 414, pp. 316–327, 1991.

[14] E. Seemann, K. Nickel, and R. Stiefelhagen, "Head pose estimation using stereo vision for human-robot interaction," in *IEEE International Conference on Automatic Face and Gesture Recognition*, May 2004, pp. 626–631.

[15] G. Fanelli, M. Dantone, J. Gall, A. Fossati, and L. Van Gool, "Random forests for real time 3d face analysis," *International Journal of Computer Vision*, vol. 101, no. 3, pp. 437–458, February 2013.

[16] X. Peng, J. Huang, Q. Hu, S. Zhang, A. Elgammal, and D. Metaxas, "From circle to 3-sphere: Head pose estimation by instance parameterization," *Computer Vision and Image Understanding*, vol. 136, pp. 92 – 102, 2015.

[17] R. S. Ghiass, O. Arandjelović, and D. Laurendeau, "Highly accurate and fully automatic head pose estimation from a low quality consumer-level rgb-d sensor," in *Proceedings of the 2nd Workshop on Computational Models of Social Interactions: Human-Computer-Media Communication*. ACM, 2015, pp. 25–34.

[18] S. Srinivasan and K. L. Boyer, "Head pose estimation using view based eigenspaces," *IEEE International Conference on Pattern Recognition*, vol. 4, pp. 302–305, 2002.

[19] B. Raytchev, I. Yoda, and K. Sakaue, "Head pose estimation by nonlinear manifold learning," in *IEEE International Conference on Pattern Recognition*, vol. 4, August 2004, pp. 462–466.

[20] N. Hu, W. Huang, and S. Ranganath, "Head pose estimation by nonlinear embedding and mapping," in *IEEE International Conference on Image Processing*, vol. 2, September 2005, pp. 342–345.

[21] Z. Li, Y. Fu, J. Yuan, T. Huang, and Y. Wu, "Query driven localized linear discriminant models for head pose estimation," in *IEEE International Conference on Multimedia and Expo*, July 2007, pp. 1810–1813.

[22] C. BenAbdelkader, "Robust head pose estimation using supervised manifold learning," in *European Conference on Computer Vision*. Springer, 2010, pp. 518–531.

[23] K. Sundararajan and D. L. Woodard, "Head pose estimation in the wild using approximate view manifolds," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2015, pp. 50–58.

[24] M. Marin-Jimenez, A. Zisserman, M. Eichner, and V. Ferrari, "Detecting people looking at each other in videos," *International Journal of Computer Vision*, vol. 106, no. 3, pp. 282–296, February 2014.

[25] A. Sharma and D. W. Jacobs, "Bypassing synthesis: Pls for face recognition with pose, low-resolution and sketch," in *IEEE Conference onComputer Vision and Pattern Recognition*. IEEE, 2011, pp. 593–600.

[26] M. Haj, J. Gonzalez, and L. Davis, "On partial least squares in head pose estimation: How to simultaneously deal with misalignment," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2012, pp. 2602–2609.

[27] P. Naik and C.-L. Tsai, "Partial least squares estimator for single-index models," *Journal of the Royal Statistical Society B*, vol. 62, no. 4, pp. 763–771, 2000.

[28] X. Zhu and D. Ramanan, "Face detection, pose estimation, and landmark localization in the wild," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2012, pp. 2879–2886.

[29] B. Ahn, J. Park, and I. S. Kweon, "Real-time head orientation from a monocular camera using deep neural network," in *Asian Conference on Computer Vision*. Springer, 2014, pp. 82–96.

[30] X. Liu, W. Liang, Y. Wang, S. Li, and M. Pei, "3D head pose estimation with convolutional neural network trained on synthetic images," in *IEEE International Conference on Image Processing*, September 2016.

[31] Z. Ghahramani and G. E. Hinton, "The EM algorithm for mixtures of factor analyzers," University of Toronto, Tech. Rep. CRG-TR-96-1, 1996.

[32] M. E. Tipping and C. M. Bishop, "Mixtures of probabilistic principal component analyzers," *Neural Computation*, vol. 11, no. 2, pp. 443–482, Feb. 1999.

[33] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, June 2001, pp. I–511–I–518 vol.1.

[34] N. Gourier, D. Hall, and J. L. Crowley, "Estimating face orientation from robust detection of salient facial features," in *IEEE International Conference on Pattern Recognition Workshop on Visual Observation of Deictic Gestures*, August 2004.

[35] M. Demirkus, J. J. Clark, and T. Arbel, "Robust semi-automatic head pose labeling for real-world face video sequences," *Multimedia Tools and Applications*, vol. 70, no. 1, pp. 495–523, 2013.

[36] M. Demirkus, D. Precup, J. J. Clark, and T. Arbel, "Hierarchical temporal graphical model for head pose estimation and subsequent attribute classification in real-world videos," *Computer Vision and Image Understanding*, vol. 136, pp. 128 – 145, 2015, generative Models in Computer Vision and Medical Imaging.

[37] R. Stiefelhagen, "Estimating head pose with neural networks – results on the pointing04 ICPR workshop evaluation data," in *IEEE International Conference on Pattern Recognition Pointing04 Workshop*, August 2004.

[38] N. Gourier, J. Maisonnasse, D. Hall, and J. L. Crowley, "Head pose estimation on low resolution images," in *Multimodal Technologies for Perception of Humans*. Springer, 2007, pp. 270–280.

[39] C. E. Rasmussen, *Gaussian processes for machine learning*. MIT Press, 2006.

[40] H. Abdi, "Partial least square regression (PLS regression)," *Encyclopedia for research methods for the social sciences*, pp. 792–795, 2003.

[41] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and computing*, vol. 14, no. 3, pp. 199–222, 2004.

[42] B. Wang, W. Liang, Y. Wang, and Y. Liang, "Head pose estimation with combined 2D SIFT and 3D HOG features," in *International Conference on Image and Graphics*, July 2013, pp. 650–655.

[43] M. Demirkus, D. Precup, J. Clark, and T. Arbel, "Soft biometric trait classification from real-world face videos conditioned on head pose estimation," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2012, pp. 130–137.

[44] M. Demirkus, D. Precup, J. J. Clark, and T. Arbel, "Probabilistic temporal head pose estimation using a hierarchical graphical model," in *European Conference on Computer Vision*, 2014, pp. 328–344.

[45] X. Xiong and F. De la Torre, "Supervised descent method and its applications to face alignment," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 532–539.

[46] B. Massé, S. Ba, and R. Horaud, "Simultaneous estimation of gaze direction and visual focus of attention for multi-person-to-robot interaction," in *IEEE International Conference on Multimedia and Expo*, Seattle, WA, Jul. 2016.

Vincent Drouard received the M.Sc. in applied mathematics from the school of engineering, university of Nice Sophia-Antipolis in 2013. He is currently a Ph.D. student at the university of Grenoble Alpes (UGA) and member of the PERCEPTION team at INRIA Grenoble Rhône-Alpes. In 2015 he received the best student paper award (second place) at the IEEE International Conference on Image Processing.

Radu Horaud received the B.Sc. degree in Electrical Engineering, the M.Sc. degree in Control Engineering, and the Ph.D. degree in Computer Science from the Institut National Polytechnique de Grenoble, France. In 1982-1984 he was a post-doctoral fellow with the Artificial Intelligence Center, SRI International, Menlo Park, CA. Currently he holds a position of director of research with INRIA Grenoble Rhône-Alpes, where he is the founder and head of the PERCEPTION team. His research interests include computer vision, machine learning, audio signal processing, audiovisual analysis, and robotics. Radu Horaud and his collaborators received numerous best paper awards. He is an area editor of the *Elsevier Computer Vision and Image Understanding*, a member of the advisory board of the *Sage International Journal of Robotics Research*, and an associate editor of the *Kluwer International Journal of Computer Vision*. He was program co-chair of IEEE ICCV'01 and of ACM ICMI'15. In 2013 Radu Horaud was awarded an ERC Advanced Grant for his project *Vision and Hearing in Action* (VHIA).

Antoine Deleforge is a tenured research scientist at INRIA with the PANAMA team, Rennes, France, since 2016. He received the B.En. (2008) and M.Sc. (2010) degrees in computer science and mathematics from ENSIMAG, Grenoble, France, and the M.Sc. (research track) degree in computer graphics, vision and robotics from the Université Joseph Fourier, Grenoble. In 2010-2013 he prepared a Ph.D in the PERCEPTION team, INRIA Grenoble Rhône-Alpes, and received the Ph.D. degree in computer science and applied mathematics in November 2013 from the university of Grenoble. In 2014-2015 he was a post-doctoral fellow in the audio group of the Multimedia Communications and Signal Processing chair of the Friedrich-Alexander university, Erlangen, Germany. His research interests are in machine learning, statistics and auditory scene analysis.

Silèye Ba received the M.Sc. (2000) in applied mathematics and signal processing from University of Dakar, Dakar, Senegal, and the M.Sc. (2002) in mathematics, computer vision, and machine learning from Ecole Normale Supérieure de Cachan, Paris, France. From 2003 to 2009 he was a PhD student and then a post-doctoral researcher at IDIAP Research Institute, Martigny, Switzerland, where he worked on probabilistic models for object tracking and human activity recognition. From 2009 to 2013, he was a researcher at Telecom Bretagne, Brest, France working on variational models for multi-modal geophysical data processing. From 2013 to 2014 he worked at RN3D Innovation Lab, Marseille, France, as a research engineer, where he used computer vision and machine learning principles and methods to develop human-computer interaction software tools. From 2014 to 2016 he was a researcher in the PERCEPTION team at INRIA Grenoble Rhône-Alpes, working on machine learning and computer vision models for human-robot interaction. Since May 2016 he is a computer vision scientist with VideoStitch, Paris.

Georgios Evangelidis is a senior computer vision scientist at Daqri International. He received his Ph.D. in Computer Engineering from the University of Patras, Patras, Greece, in 2008. He joined the Multimedia Pattern Recognition group at Fraunhofer IAIS, Bonn, Germany as a post-doctoral researcher (2009-2010) and the PERCEPTION team at INRIA Grenoble Rhône-Alpes, France as a researcher (2012-2016). His expertise includes 3D geometry, depth sensors, action recognition and SLAM.